**In the Claims:**

Please amend claims 1, 15, and 26 as indicated below.

1. (Currently amended) A system, comprising:

a processor; and

memory coupled to the processor and configured to store program instructions executable by the processor to implement a software documentation generator configured to:

input a plurality of sources related to a software program, wherein the plurality of sources comprises a plurality of different types of sources and comprises one or more of a software library documentation file for the software program and software program source code for the software program~~;~~

analyze the ~~one or more~~ plurality of sources to identify a type of each of the sources;

extract information from the plurality of sources based on the type of the source, wherein the software documentation generator includes a plurality of different input source plug-ins, wherein each input source plug-in corresponds to a respective one of the source types and each input source plug-in is configured to extract information from sources of a type to which the plug-in corresponds;

aggregate the extracted information into a uniform format; and

transform the aggregated information into one or more specified sets of software documentation for the software program, wherein the software documentation generator includes a plurality of different transformer plug-in sets, wherein each transformer plug-in set corresponds to one or more respective types of output software documentation sets and each transformer plug-in set is configured to generate one or more ~~output~~ respective output software documentation sets of types to which the plug-in corresponds, wherein the specified sets of software documentation document the functionality of the software program.

2. (Previously presented) The system as recited in claim 1, wherein one of the sources comprises the software library documentation file.

3. (Original) The system as recited in claim 2, wherein the software library documentation file comprises a tag library descriptor (TLD).

4. (Previously presented) The system as recited in claim 1, wherein one of the sources comprises the software program source code.

5. (Original) The system as recited in claim 1, wherein the documentation sets comprise documentation for one or more application programming interfaces (API) provided by a software library.

6. (Canceled)

7. (Previously presented) The system as recited in claim 1, wherein an input source plug-in is configured to generate information not included in the corresponding source file.

8. (Previously presented) The system as recited in claim 1, wherein each input source plug-in is configured to output data in the uniform aggregate format.

9. (Canceled)

10. (Previously presented) The system as recited in claim 1, wherein the input source plug-ins are configured to produce a uniformly formatted aggregate input document and wherein each transformer plug-in set is configured to input data included in the uniformly formatted aggregate input document.

11. (Original) The system as recited in claim 10, wherein a transformer plug-in set is configured to generate information not included in the uniformly formatted aggregate input document.

12. (Original) The system as recited in claim 1, wherein the output software documentation sets comprise one or more text files.

13. (Original) The system as recited in claim 1, wherein the output software documentation sets comprise one or more portable document files (PDF).

14. (Original) The system as recited in claim 1, wherein the output software documentation sets comprise one or more hypertext markup language (HTML) files.

15. (Currently amended) A method, comprising

receiving information from multiple sources related to a software program, wherein the multiple sources comprise a plurality of different types of sources and comprise one or more of a software library documentation file for the software program and software program source code for the software program;

extracting documentation data from said sources, wherein said extracting comprises:

analyzing a source for type and data format; and

selecting a corresponding one of a plurality of different input source plug-ins based on said analyzing to extract the documentation data;

aggregating the extracted documentation data in a uniform format; and

transforming the uniformly formatted documentation data into one or more specified documentation sets for the software program, wherein the one or more specified sets of software documentation document the functionality of the software program, and wherein said transforming comprises:

selecting one of a plurality of transformer plug-in sets corresponding to a specified output documentation format; and

the selected transformer plug-in set translating a portion of the uniformly formatted aggregate data into one or more elements of a software documentation set in the specified output documentation format.

16. (Previously presented) The method as recited in claim 15, wherein one of the sources comprises the software library documentation file.

17. (Original) The method as recited in claim 16, wherein the software library documentation file comprises a tag library descriptor (TLD).

18. (Previously presented) The method as recited in claim 15, wherein one of the sources comprises the software program source code.

19. (Previously presented) The method as recited in claim 15, wherein the documentation sets comprise documentation for one or more application programming interfaces (APIs) provided by a software library.

20. (Canceled)

21. (Original) The method as recited in claim 15, wherein said aggregating comprises:

if a uniformly formatted aggregate input document specifies information not comprised in data extracted from the source, generating said information; and

generating a uniformly formatted aggregate input document.

22. (Canceled)

23. (Original) The method as recited in claim 15, wherein the output software documentation sets comprise one or more text files.

24. (Previously presented) The method as recited in claim 15, wherein the output software documentation sets comprise one or more portable document files (PDFs).

25. (Previously presented) The method as recited in claim 15, wherein the output software documentation sets comprise one or more hypertext markup language (HTML) files.

26. (Currently amended) A computer-accessible memory medium storing program instructions, wherein the program instructions are computer-executable to:

input a plurality of sources related to a software program, wherein the plurality of sources comprises a plurality of different types of sources and comprises one or more of a software library documentation file for the software program and software program source code for the software program;

analyze the one or more source files to identify the type of each of the source files;

extract information from the plurality of sources based on the type of the source, wherein to extract comprises to:

analyze a source for type and data format; and

select a corresponding one of a plurality of input source plug-ins based on said analyzing to extract the information from the source;

aggregate the extracted information into a uniform format; and

transform the aggregated information into one or more specified sets of software documentation for the software program, wherein the one or more specified sets of software documentation document the functionality of the software program, and wherein to transform comprises to:

select one of a plurality of transformer plug-in sets corresponding to a specified output documentation format; and

translate a portion of the uniformly formatted aggregate information into one or more elements of a software documentation set in the specified output documentation format.

27. (Previously presented) The computer-accessible medium as recited in claim 26, wherein one of the sources comprises the software library documentation file.

28. (Previously presented) The computer-accessible medium as recited in claim 27, wherein the software library documentation file comprises a tag library descriptor (TLD).

29. (Previously presented) The computer-accessible medium as recited in claim 26, wherein one of the sources comprises the software program source code.

30. (Previously presented) The computer-accessible medium as recited in claim 26, wherein the documentation sets comprise documentation for one or more application programming interfaces (APIs) provided by a software library.

31. (Canceled)

32. (Original) The computer-accessible medium as recited in claim 26, wherein to aggregate comprises to:

if a uniformly formatted aggregate input document specifies information not comprised in data extracted from the source, generate said information; and

generate the uniformly formatted aggregate input document.

33. (Canceled)

34. (Original) The computer-accessible medium as recited in claim 26, wherein the output software documentation sets comprise one or more text files.

35. (Previously presented) The computer-accessible medium as recited in claim 26, wherein the output software documentation sets comprise one or more portable document files (PDFs).

36. (Previously presented) The computer-accessible medium as recited in claim 26, wherein the output software documentation sets comprise one or more hypertext markup language (HTML) files.